REMARKS

This communication is in response to the Office Action mailed May 17, 2007. In the Office Action, claims 1-11 are rejected as being unpatentable over Maroulis in view of Ford and Roger. Claims 12-13 and 14-15 are rejected as being unpatentable over Maroulis, Ford and Roger as applied to claims 1 and 5, and further in view of Petty.

TELEPHONIC INTERVIEW

Applicant appreciates the time that Examiner took to conduct a telephonic interview with the undersigned on July 23, 2007. Also present during the telephonic interview was Karl Ginter, who is a technical consultant of the assignee. During the interview, the following two related applications were discussed -- 10/086,602 and 10/086,268.

No exhibits were shown nor were any demonstrations conducted. No particular claims were discussed. However, the feature of an "enterprise directory" in the claims, including the enterprise directory being "a directory of named objects, including users, network devices and network services" was discussed.

The specific prior art that was discussed includes U.S. Patent 6,909,708 to Krishnaswamy et al. and assigned (on its face) to MCI Communications Corporation. This prior art has been applied in the related application (10/086,602).

No proposed amendments were discussed.

The principal argument of the Applicant is that the "database" disclosed by the Krishnaswamy patent does not anticipate the "enterprise directory" feature recited in the claims. The Examiner argued that Applicant's specification does not distinguish between a "database" and an "enterprise directory" and that, therefore, the Examiner is free to apply the "database" of Krishnaswamy against the "enterprise directory" recited in the claims.

It was understood during the interview that Krishnaswamy was being considered "representative" with respect to its disclosure of a "database" and that the discussion likewise applied to references being applied in the present application, with respect to their respective disclosures of a "database."

No agreement was reached during the interview.

Prior-Art-Based Rejections

In response to the previous Office Action, Applicant amended the independent claims to recite enterprise directory information is integrated into the operation of the CTI control mechanism with respect to a particular user, wherein the enterprise wide directory[1] is a directory of named objects, including users, network devices and network services. Applicant referred the Examiner to, "for example, [0085] and [0086] of the published version of Applicant's patent application as support for this amendment."

Applicant further stated that

> The independent claims now recite that the user CTI control mechanism has "an interface via which each of a plurality of particular users can configure a CTI application to logically associate a computer and a gateway telephone in physical proximity to the computer with the telephonic identity of that particular user." Dependent claims 12-15 have been added to recite additional features of the "interface."

Applicant further argued that none of Maroulis, Ford or Dekelbaum recite that enterprise wide directory information is integrated into the operation of the CTI control mechanism with respect to a particular user, wherein the enterprise wide directory is a directory of named objects, including users, network devices and network services. In the present rejection, the Examiner now cites the Roger reference and does not cite the Dekelbaum reference.

Applicant will discuss the following in some detail:

- "Enterprise directory" cannot be properly construed to include a "database" as disclosed in the cited references.
- The Examiner is incorrect in the requirement, stated during the interview and also stated in an Advisory Action in the related 10/086,602 application, that Applicant's specification must show the difference between a "database" and an "enterprise directory."[2]

"Enterprise directory" cannot be properly construed to include a "database" as disclosed in the cited references

In this discussion, Applicant repeats and incorporates the discussion made on this point in the related application 10/086,602, in the Amendment C filed on August 14, 2007. Although, in

[1] Applicant now further slightly amends the claims, where appropriate, such that the claims consistently use the term "enterprise directory" (as opposed to "enterprise-wide directory") which, as will be discussed in greater detail below, has an ordinary and accustomed meaning. This amendment is supported throughout the specification and also incorporates the terminology used in the related application 10/086,602.

[2] In the Advisory Action in the related 10/086,602 application, the Examiner's language is conclusory that there is no difference: "In reply, the directory data of the prior art and the enterprise directory of the applicant are not different because the specification of the application does not show the different [sic] between the databases."

the related application 10/086,602, the discussion was specifically with respect to the Krishnaswamy reference cited in that application, the discussion is generally applicable in the present application with respect to the Ford and Roger references cited herein.

In particular, Applicant provides herewith evidence that the "enterprise directory" recited in the claims is not covered by the Ford and Roger "databases." Applicant does this by way of providing several references from about the time of Applicant's priority date.

1. T. Howes and M. Smith, LDAP: Programming Directory Enabled Applications with Lightweight Directory Access Protocol. Macmillan Technology Series, 1997. In addition to the cover and publication pages, Applicant provides herewith an excerpt at pages 4-5. See Appendix A to this Amendment C. In the excerpt, in the section entitled "What a Directory Service is Not," it is clearly stated that a directory services is not a general purpose database. While this entire section is provided in Appendix A, Applicant would like to draw the Examiner's attention to one particular paragraph:

> The first thing you should not do is treat the directory like a general database. It's not designed with that kind of use in mind, and you'll likely be disappointed with the results. This means the directory is not suited to provide transaction-based service, cannot generally handle huge numbers of updates, and usually would make a bad engine for an SQL application.

2. United States Patent No. 6,016,499, assigned on its face to Novell, Inc. The filing date of the priority provisional application is February 20, 1997, and the filing date of the non-provisional application is July 21, 1997. At col. 1, line 60 et seq, for example, the applicant distinguishes between "directory services" and "relational databases."

> Unlike relational databases, "directory service" structures are a relatively recent development in information technology. The need for directory services became clear only after computer networks became a prominent part of the information landscape and grew so large and complex that their administration required full-time attention from specialists. Directory services are sometimes referred to as "naming services."
>
> A variety of directory service providers are now available to help administer both the location of network resources and the rights of network users to use those resources. Many, but not all, directory service tools conform at least in part with the X.500 directory services standard. One well-known directory service system includes NetWare Directory Services software commercially available from Novell, Inc. of Provo, Utah (NETWARE DIRECTORY SERVICES is a trademark of Novell, Inc.). As used herein, "Novell Directory Services" ("NDS") includes NetWare Directory Services and any other directory service commercially available from Novell, Inc.
>
> In contexts other than the present one, a directory services repository is sometimes called a directory services "database." Both repositories and databases may be distributed,

because that is mainly a matter of storage and locks for enforcing consistency, rather than a question of the basic internal structure. But "database" and "repository" are not interchangeable in the present context.

A repository supports naming services. Each repository is organized hierarchically, as one or more trees. Each object in a tree (except the root object) has exactly one parent. Objects may generally have children, which may inherit properties from their ancestor objects. Objects are instances of "classes," as described in detail below.

By contrast, "database" as used herein refers to a relational database. A given database may support any of a wide variety of commercial or personal activities. Each database is organized as a set of tables in which rows represent records and columns represent record fields. Certain fields may be found in multiple tables, and the values of these "key" or "index" fields are used to guide database searches. Database access and manipulation involve combining information from various tables into new combinations to obtain different views of the data.

In short, databases and repositories arose at different times to meet different needs, and have different structures and capabilities. There is no need to dwell further here on the details of relational database structure, SQL, or ODBC, as numerous references on those topics are widely available. Many computer science professionals have also taken at least one course in databases.

3. The SLAPD and SLURPD Administrator's Guide, University of Michigan, 30 April 1996, Release 3.3. In addition to the cover, publication pages and Table of Contents, Applicant provides herewith an excerpt at pages 1-10. See Appendix B to this Amendment C. In the excerpt, in the section entitled "1.1 What is a directory service?" it is discussed how a directory service differs from a database. The entire section 1.1 (two paragraphs only), is reproduced below:

1.1 What is a directory service?
A directory is like a database, but tends to contain more descriptive, attribute-based information. The information in a directory is generally read much more often than it is written. As a consequence, directories don't usually implement the complicated transaction or roll-back schemes regular databases use for doing high-volume complex updates. Directory updates are typically simple all-or-nothing changes, if they are allowed at all. Directories are tuned to give quick-response to high-volume lookup or search operations. They may have the ability to replicate information widely in order to increase availability and reliability, while reducing response time. When directory information is replicated, temporary inconsistencies between the replicas may be OK, as long as they get in sync eventually.

There are many different ways to provide a directory service. Different methods allow different kinds of information to be stored in the directory, place different requirements on how that information can be referenced, queried and updated, how it is protected from unauthorized access, etc. Some directory services are local, providing service to a restricted context (e.g., the finger service on a single machine). Other services are global, providing service to a much broader context (e.g., the entire Internet). Global services are

> usually distributed, meaning that the data they contain is spread across many machines, all of which cooperate to provide the directory service. Typically a global service defines a uniform namespace which gives the same view of the data no matter where you are in relation to the data itself.

Applicant has thus amply demonstrated that the "enterprise directory" recited in the claims is not covered by the "databases" disclosed in Ford and Roger. Since the Examiner's contention regarding the "enterprise directory" pervades all the prior art based rejections, it is respectfully submitted that all of the prior art based rejections are improper and should be withdrawn.

The Examiner is Incorrect in the Requirement, Stated During the Interview and also Stated in an Advisory Action in the Related 10/086,602 Application, that Applicant's Specification Show the Difference between a "Database" and an "Enterprise Directory."

We now discuss the Examiner's contention that Applicant's specification must show the difference between a "database" and an "enterprise directory." As mentioned above, this contention was made in both the Examiner interview in the present application and related application (10/086,602), as well as in the advisory action in the related application.

We specifically refer the Examiner to MPEP 2111.01, which discusses the subject of ascribing the "plain meaning" to claim terms. In Amendment C in the related application 10/086,602, Applicant pointed out, without extensive discussion or support, that:

> It is respectfully submitted that there is no such requirement for Applicant's specification to distinguish between a "database" and an "enterprise directory." If this was the case, then applicants would be required to predict every rejection an Examiner may present and provide rebuttal distinguishing language in the specification, even before the patent application has been filed. Clearly, this cannot be the requirement.

Given that the Examiner continues to require (e.g., in the Advisory Action in the related application 10/086,602) that Applicant's specification distinguish between a "database" and an "enterprise directory," Applicant now discusses in greater detail the concept of "plain meaning" and "ordinary and customary meaning" attributed to a term by "those of ordinary skill in the art." More particularly, Applicant discusses MPEP 2111.01 and some Federal Circuit cases on the same point, as rebuttal to the Examiner's requirement that Applicant's specification show the difference between a database as disclosed in Ford and Roger and an enterprise directory as recited in the claims.

For example, MPEP 2111.01 quotes the Phillips v. AWH case, reciting "[T]he ordinary and customary meaning of a claim term is the meaning the term would have to a person of ordinary skill in the art in question at the time of the invention, *i.e.*, as of the effective filing date of the patent application." MPEP 2111.01 also quotes the Brookhill-Wilk 1, LLC, v. Intuitive Surgical, Inc. case, reciting "In the absence of an express intent to impart a novel meaning to the claim terms, the words are presumed to take on the ordinary and customary meaning attributed to them by those of ordinary skill in the art." As a last example, MPEP 2111.01 also quotes the ACTV, Inc. v. The Walt Disney Company case, which recites "Since there was no express definition for the term 'URL' in the specification, the term should be given its broadest reasonable interpretation consistent with the intrinsic record and take on the ordinary and customary meaning attributed to it by those of ordinary skill in the art…."

Thus, MPEP 2111.01 and the cases cited therein implore the Examiner to use the "ordinary and customary meaning" attributed to the term "enterprise directory" *absent* some indication on the part of Applicant to impart a novel meaning to the term. Put another way, unless the Applicant intends for the term "enterprise directory" to have a meaning other than the "ordinary and customary" meaning, there is no requirement for Applicant to explicitly define the term "enterprise directory" in the specification and/or distinguish the term from "database" as used, for example, in Ford and Roger.

As Applicant has pointed out in great detail, the "ordinary and customary meaning" attributed to the term "enterprise directory" is not covered by a database and, in fact, explicitly disclaims the notion of being covered by a database. The Examiner has refused, in the related application 10/086,602 to consider Applicant's provided evidence of such, instead insisting that Application should have foreseen the argument the Examiner would make regarding the term "database."

Perhaps the Examiner is mixing in the doctrine of "A patentee may be his own lexicographer." In accordance with that doctrine, MPEP 2111.01 cites the In re Paulsen case, reciting that (emphasis added) "inventor may define specific terms used to describe invention, but must do so 'with reasonable clarity, deliberateness, and precision' and, if done, must "set out his uncommon definition in some manner within the patent disclosure "'so as to give one of ordinary skill in the art notice of the change' in meaning." However, this requirement for the Applicant to define specific terms in the specification does not apply where the Applicant is not making an "uncommon definition" to change the meaning of terms from what would be the ordinary and customary meaning. That is, Applicant is not changing the meaning of the term "enterprise directory" from what is the ordinary and customary meaning as evidenced, for example, by the various references cited to the Examiner above and provided herein in the appendices.

Given all this, perhaps the Examiner does in fact believe that a reasonable interpretation of "enterprise directory" is broad enough to be covered by "database" as used by Ford and Roger. MPEP 2111.01 discusses, again referring to the Phillips v. AWH case, "The ordinary and customary meaning of a term may be evidenced by a variety of sources, including 'the words of the claims themselves, the remainder of the specification, the prosecution history, and extrinsic evidence concerning relevant scientific principles, the meaning of technical terms, and the state of the art.'" Of course, as discussed above, Applicant believes that all these sources, and particularly the extrinsic evidence, make clear, contrary to what may be the Examiner's belief, that the ordinary and customary meaning of "enterprise directory" is not broad enough to be covered by the Ford and Roger "databases."

Applicant recognizes that, in some circumstances, it may be proper to require a patent specification to distinguish among two or more meanings of a claim term. In the correct context, this requirement would be limited, though, to the circumstance referenced in MPEP 2111.01 in which the evidence of ordinary and customary meaning is ambiguous. This is not the case here. Referring to the Brookhill-Wok case, MPEP 2111.01 states "If extrinsic sources, such as dictionaries, evidence more than one definition for the term, the intrinsic record must be consulted to identify which of the different possible definitions is most consistent with applicant's use of the terms." Also, referring to the Renishaw plc v. Marposs Societa' per Azioni case, MPEP 2111.01 states "Where there are several common meanings for a claim term, the patent disclosure serves to point away from the improper meanings and toward the proper meanings."

Applicant has pointed out several items of extrinsic evidence to support the definition of "enterprise directory" as distinct from a database such as disclosed by Ford and Roger. If the Examiner continues to require Applicant's specification to explicitly define the "enterprise directory" claim term, then Applicant respectfully requests the Examiner to point out evidence of another ordinary and customary meaning of "enterprise directory" that may include a database such as the Ford and Roger "databases."

In the absence of such a showing by the Examiner, it is respectfully submitted that:

- The ordinary and customary meaning of "enterprise directory" is unambiguous, as evidenced by the several items of extrinsic evidence pointed out by Applicant;

- Since the ordinary and customary meaning of "enterprise directory" is unambiguous, there is no independent requirement for Applicant's specification to identify a particular definition used by Applicant.
- The ordinary and customary meaning of "enterprise directory" cannot be construed to include a "database" such as the Ford and Roger databases.
- The claims are patentable over the combination of Maroulis, Ford and Roger (and Petty as well, as applied against claims 12-13 and 14-15).

CONCLUSION

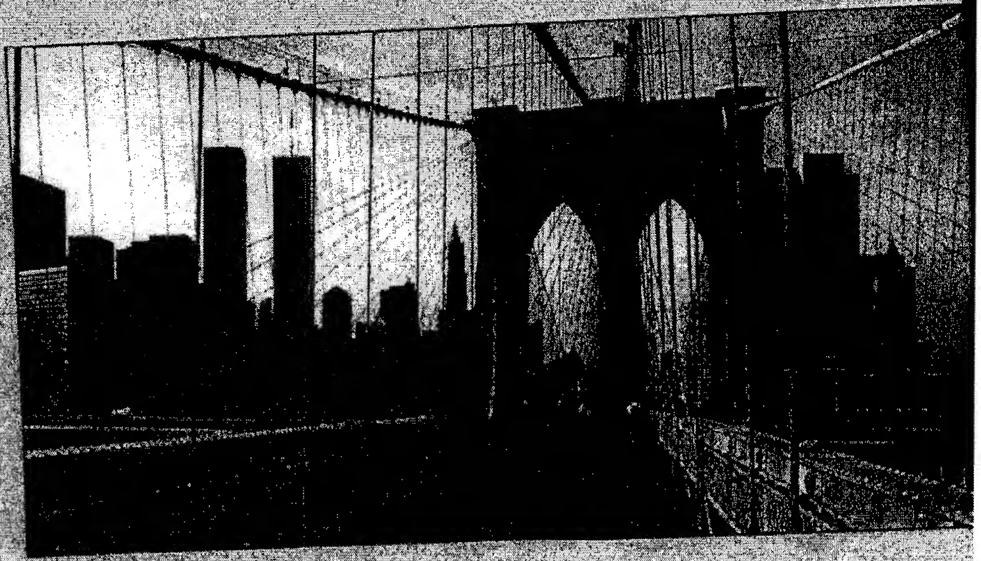
Applicants believe that all pending claims are allowable and respectfully request a Notice of Allowance for this application from the Examiner. Should the Examiner believe that a telephone conference would expedite the prosecution of this application, the undersigned can be reached at the telephone number set out below.

Respectfully submitted,
BEYER WEAVER LLP

/ASH/
Alan S. Hodes
Reg. No. 38,185

P.O. Box 70250
Oakland, CA 94612-0250
408-255-8001

APPENDIX A



Macmillan Technology Series

LDAP

Programming Directory-Enabled Applications with Lightweight Directory Access Protocol

Timothy A. Howes, Ph.D.
Mark C. Smith

LDAP:

Programming Directory-Enabled Applications with Lightweight Directory Access Protocol

Timothy A. Howes

Mark C. Smith

MTP
MACMILLAN TECHNICAL PUBLISHING U·S·A

Macmillan Technical Publishing, Indianapolis, Indiana

LDAP: Programming Directory-Enabled Applications with Lightweight Directory Access Protocol

By Timothy A. Howes and Mark C. Smith

Published by:
Macmillan Technical Publishing
201 West 103rd Street
Indianapolis, IN 46290 USA

Printed in the United States of America 4 5 6 7 8 9 0

Library of Congress Catalog Card Number: 96-78510

ISBN: 1-57870-000-0

Warning and Disclaimer

This book is designed to provide information about the LDAP computer program. Every effort has been made to make this book as complete and as accurate as possible, but no warranty or fitness is implied.

The information is provided on an "as is" basis. The author(s) and Macmillan Technical Publishing shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book or from the use of the disks or programs that may accompany it.

Publisher	*Don Fowley*
Publisher's Assistant	*Rosemary Lewis*
Publishing Manager	*Jim LeValley*
Marketing Manager	*Kourtnaye Sturgeon*
Managing Editor	*Carla Hall*

Acquisitions Editor
Jim LeValley

Senior Editors
Sarah Kearns
Suzanne Snyder

Development Editor
Tim Huddleston

Project/Copy Editor
Mitzi Foster

Acquisitions Coordinator
Amy Lewis

Cover Designer
Sandra Schroeder

Cover Production
Aren Howell

Book Designer
Sandra Schroeder

Manufacturing Coordinator
Brook Farling

Production Manager
Kelly Dobbs

Production Team Supervisors
Laurie Casey, Joe Millay

Graphics Image Specialists
Dan Harris
Laura Robbins
Dennis Sheehan

Production Analyst
Erich J. Richter

Production Team
Trina Brown, Tricia Flodder, Christopher Morris, Megan Wade

Indexer
Eric Brinkman

While directories come in all shapes and sizes, most examples share some common elements that bind them together. Directories are special purpose databases, usually containing typed information. Read access to the directory is typically much more frequent than update access. Some directories do not allow updates at all, although LDAP does. Directory services are distinguished from name services by their ability to search for, as well as retrieve, named information. They are distinguished from general databases by their higher access/update ratio, their lack of transaction semantics, and their limited ability to separate search from retrieval. The concept of "result set," so useful in a general database system, is not often found in a directory service.

Directories may be replicated for improved reliability and performance, with either strong or weak consistency requirements. Directories may include authentication and access control capabilities, or they may only provide world-readable information that does not require such protection.

Many directories, including LDAP-based ones, essentially provide a way to name, manage, and access collections of attribute-value pairs. Where directories differ from one another is in the way in which this information is represented and accessed, the flexibility with which the information can be searched, whether the kinds of information in the directory may be extended, and whether and how the information can be updated.

The preceding paragraphs provide a description of a directory service, not a formal definition. It is intentionally broad, though we will narrow our definition later when we focus in on LDAP directory service, the subject of this book.

What a Directory Service Is Not

Having a directory service capable of storing arbitrary attribute-value pairs is an attractive thing. So attractive, in fact, that you may be tempted to put some things in it that you should not, or to use the directory for purposes for which it is not well suited. So the question naturally arises, "What should you *not* do with a directory service?"

The first thing you should not do is treat the directory like a general database. It's not designed with that kind of use in mind, and you'll likely be disappointed with the results. This means the directory is not suited to provide transaction-based service, cannot generally handle huge numbers of updates, and usually would make a bad engine for an SQL application.

Second, a directory is not a file system. Storing all the files on your system in the directory is probably a bad idea. In fact, any time you are thinking of storing very large objects in the directory, you should ask yourself whether it might not be more appropriate to store only a pointer to the object in the directory, and the object itself in a service more suited to storing and retrieving large objects (for example, on an FTP server or in a file system).

Finally, a directory should not be used unless it's providing your application some benefit. Does the information you want to store in the directory need to be accessed by others? By you or your application from different locations? If it is only to be accessed by one application on a specific host, then there is little benefit derived from storing it in the directory, yet there could be overhead, added complexity, and other factors that argue for a local solution.

Directory-Enabled Applications

For a few applications—those that read, write, or manage directory information as their primary purpose—the benefit of directory-enabling is clear. The applications could not exist without it, after all. But for the vast majority of applications, directory support is ancillary to the application's main purpose, which might be browsing the World Wide Web; sending, receiving, and processing e-mail; organizing information on your intranet; or just about anything else you can think of. It is this latter category of applications that represents the vast majority of potential directory client applications that might benefit from being *directory-enabled*.

APPENDIX B

The SLAPD and SLURPD Administrator's Guide

University of Michigan

30 April 1996
Release 3.3

Copyright

Acknowledgments

The LDAP development team at the University of Michigan consists of Tim Howes, Mark Smith, Gordon Good, Lance Sloan and Steve Rothwell. Our thanks also to Bryan Beecher, Frank Richter, Eric Rosenquist, Peter Whittaker, Martijn Koster, Craig Watkins, Rocky Rakesh Patel, Alan Young, Mark Prior, Enrique Silvestre Mora, Roland Hedberg, and numerous others.

Table of Contents

1. Introduction to *slapd* and *slurpd*

This document describes how to build, configure, and run the stand-alone LDAP daemon (*slapd*) and the stand-alone LDAP update replication daemon (*slurpd*). It is intended for newcomers and experienced administrators alike. This section provides a basic introduction to directory service, and the directory service provided by *slapd* in particular.

1.1 What is a directory service?

A directory is like a database, but tends to contain more descriptive, attribute-based information. The information in a directory is generally read much more often than it is written. As a consequence, directories don't usually implement the complicated transaction or roll-back schemes regular databases use for doing high-volume complex updates. Directory updates are typically simple all-or-nothing changes, if they are allowed at all. Directories are tuned to give quick-response to high-volume lookup or search operations. They may have the ability to replicate information widely in order to increase availability and reliability, while reducing response time. When directory information is replicated, temporary inconsistencies between the replicas may be OK, as long as they get in sync eventually.

There are many different ways to provide a directory service. Different methods allow different kinds of information to be stored in the directory, place different requirements on how that information can be referenced, queried and updated, how it is protected from unauthorized access, etc. Some directory services are *local*, providing service to a restricted context (e.g., the finger service on a single machine). Other services are global, providing service to a much broader context (e.g., the entire Internet). Global services are usually *distributed*, meaning that the data they contain is spread across many machines, all of which cooperate to provide the directory service. Typically a global service defines a uniform *namespace* which gives the same view of the data no matter where you are in relation to the data itself.

1.2 What is LDAP?

Slapd's model for directory service is based on a global directory model called LDAP, which stands for the Lightweight Directory Access Protocol. LDAP is a directory service protocol that runs over TCP/IP. The nitty-gritty details of LDAP are defined in RFC 1777 "The Lightweight Directory Access Protocol." This section gives an overview of LDAP from a user's perspective.

What kind of information can be stored in the directory? The LDAP directory service model is based on *entries*. An entry is a collection of *attributes* that has a name, called a *distinguished name* (DN). The DN is used to refer to the entry unambiguously. Each of the entry's attributes has a *type* and one or more *values*. The types are typically mnemonic strings, like "`cn`" for common name, or "`mail`" for email address. The values depend on what type of attribute it is. For example, a `mail` attribute might contain the value "`babs@umich.edu`". A `jpegPhoto` attribute would contain a photograph in binary JPEG/JFIF format.

How is the information arranged? In LDAP, directory entries are arranged in a hierarchical tree-like structure that reflects political, geographic and/or organizational boundaries. Entries representing countries appear at the top of the tree. Below them are entries representing states or national organizations. Below them might be entries representing people, organizational units, printers, documents, or just about anything else you can think of. Figure 1 shows an example LDAP directory tree, which should help make things clear.

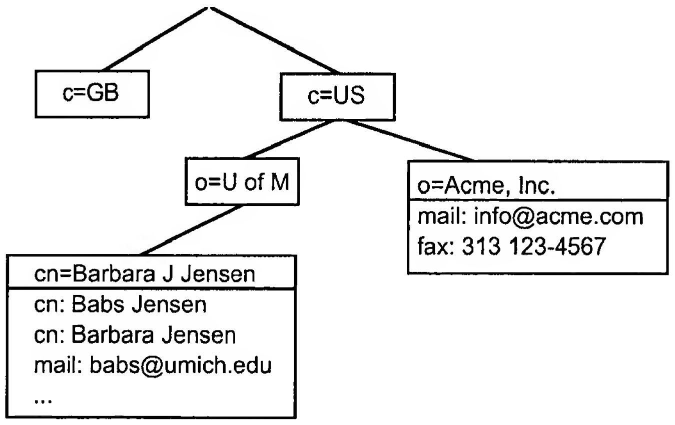


Figure 1: An example LDAP directory tree.

In addition, LDAP allows you to control which attributes are required and allowed in an entry through the use of a special attribute called `objectclass`. The values of the `objectclass` attribute determine the *schema rules* the entry must obey.

How is the information referenced? An entry is referenced by its distinguished name, which is constructed by taking the name of the entry itself (called the relative distinguished name, or RDN) and concatenating the names of its ancestor entries. For example, the entry for Barbara Jensen in the example above has an RDN of "`cn=Barbara J Jensen`" and a DN of "`cn=Barbara J Jensen, o=U of M, c=US`". The full DN format is described in RFC 1779, "A String Representation of Distinguished Names."

How is the information accessed? LDAP defines operations for interrogating and updating the directory. Operations are provided for adding and deleting an entry from the directory, changing an existing entry, and changing the name of an entry. Most of the time, though, LDAP is used to search for information in the directory. The LDAP search operation allows some portion of the directory to be searched for entries that match some criteria specified by a search filter. Information can be requested from each entry that matches the criteria.

For example, you might want to search the entire directory subtree below the University of Michigan for people with the name Barbara Jensen, retrieving the email address of each entry found. LDAP lets you do this easily. Or you might want to search the entries directly below the c=US entry for organizations with the string "Acme" in their name, and that have a fax number. LDAP lets you do this too. The next section describes in more detail what you can do with LDAP and how it might be useful to you.

How is the information protected from unauthorized access? Some directory services provide no protection, allowing anyone to see the information. LDAP provides a method for a client to authenticate, or prove its identity to a directory server, paving the way for rich access control to protect the information the server contains.

1.3 How does LDAP work?

LDAP directory service is based on a *client-server* model. One or more LDAP servers contain the data making up the LDAP directory tree. An LDAP client connects to an LDAP server and asks it a question. The server responds with the answer, or with a pointer to where the client can get more information (typically, another LDAP server). No matter which LDAP server a client connects to, it sees the same view of the directory; a name presented to one LDAP server references the same entry it would at another LDAP server. This is an important feature of a global directory service, like LDAP.

1.4 What is slapd and what can it do?

Slapd is an LDAP directory server that runs on many different UNIX platforms. You can use it to provide a directory service of your very own. Your directory can contain pretty much anything you want to put in it. You can connect it to the global LDAP directory service, or run a service all by yourself. Some of *slapd*'s more interesting features and capabilities include:

Choice of databases: *Slapd* comes with three different backend databases you can choose from. They are LDBM, a high-performance disk-based database; SHELL, a database interface to arbitrary UNIX commands or shell scripts; and PASSWD, a simple password file database.

Multiple database instances: *Slapd* can be configured to serve multiple databases at the same time. This means that a single *slapd* server can respond to requests for many logically different portions of the LDAP tree, using the same or different backend databases.

Generic database API: If you require even more customization, *slapd* lets you write your own backend database easily. *Slapd* consists of two distinct parts: a front end that handles protocol communication with LDAP clients; and a backend that handles database operations. Because these two pieces communicate via a well-defined C API, you can write your own customized database backend to *slapd.*

Access control: *Slapd* provides a rich and powerful access control facility, allowing you to control access to the information in your database(s). You can control access to entries based on LDAP authentication information, IP address, domain name and other criteria.

Threads: *Slapd* is threaded for high performance. A single multi-threaded *slapd* process handles all incoming requests, reducing the amount of system overhead required. *Slapd* will automatically select the best thread support for your platform.

Replication: *Slapd* can be configured to maintain replica copies of its database. This master/slave replication scheme is vital in high-volume environments where a single *slapd* just doesn't provide the necessary availability or reliability.

Configuration: *Slapd* is highly configurable through a single configuration file which allows you to change just about everything you'd ever want to change. Configuration options have reasonable defaults, making your job much easier.

Slapd also has its limitations, of course. It does not currently handle aliases, which are part of the LDAP model. The main LDBM database backend does not handle range queries or negation queries very well. These features and more will be coming in a future release.

1.5 What about X.500?

LDAP was originally developed as a front end to X.500, the OSI directory service. X.500 defines the Directory Access Protocol (DAP) for clients to use when contacting directory servers. DAP is a heavyweight protocol that runs over a full OSI stack and requires a significant amount of computing resources to run. LDAP runs directly over TCP and provides most of the functionality of DAP at a much lower cost.

This use of LDAP makes it easy to access the X.500 directory, but still requires a full X.500 service to make data available to the many LDAP clients being developed. As with full X.500 DAP clients, a full X.500 server is no small piece of software to run.

The stand-alone LDAP daemon, or *slapd*, is meant to remove much of the burden from the server side just as LDAP itself removed much of the burden from clients. If you are already running an X.500 service and you want to continue to do so, you can probably stop reading this guide, which is all about running LDAP via *slapd*, without running X.500. If you are not running X.500, want to stop running X.500, or have no immediate plans to run X.500, read on.

It is possible to replicate data from a *slapd* directory server to an X.500 DSA, which allows your organization to make your data available as part of the global X.500 directory service on a "read-only" basis. This is discussed in section 11.6.

Another way to make data in a *slapd* server available to the X.500 community would be by using a X.500 DAP to LDAP gateway. At this time, no such software has been written (to the best of our knowledge), but hopefully some group will see fit towrite such a gateway.

1.6 What is slurpd and what can it do?

Slurpd is a UNIX daemon that helps *slapd* provide replicated service. It is responsible for distributing changes made to the master *slapd* database out to the various *slapd* replicas. It frees *slapd* from having to worry that some replicas might be down or unreachable when a change comes through; *slurpd* handles retrying failed requests automatically. *Slapd* and *slurpd* communicate through a simple text file that is used to log changes.